Data Augmentation in Overcoming Catastrophic Forgetting

Zhuokai Zhao
University of Chicago
Chicago, Illinois
zhuokai@uchicago.edu

Ping-Jung Liu
University of Chicago
Chicago, Illinois
pliu5@uchicago.edu

Nanqinqin Li
University of Chicago
Chicago, Illinois
linanqinqin@uchicago.edu

ABSTRACT

With the emergence of transfer learning, there have been numerous research projects that worked toward mitigating catastrophic forgetting of deep neural network, in which neural networks tend to forget previously learned tasks after sequentially learning different formats of inputs. The paper surveys several existing methods by applying the techniques to a new benchmark dataset, Fashion-MNIST[2]. Various data augmentation methods and new training schemes, namely datasets mixing and review, have been proposed and tested on top of the existing methods with the new dataset. As a result, further improvements in overcoming catastrophic forgetting have been observed. The paper also investigated the system-level performance of the model by running with different hardware.

CCS CONCEPTS

• **Computing methodologies → Learning settings**; • **General and reference** → *Evaluation*;

KEYWORDS

Catastrophic forgetting

ACM Reference Format:
Zhuokai Zhao, Ping-Jung Liu, and Nanqinqin Li. 2018. Data Augmentation in Overcoming Catastrophic Forgetting. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In this paper, we surveyed two existing methods that are aimed to mitigate catastrophic forgetting of deep neural networks, synaptic intelligence (SI)[4] and elastic weight consolidation (EWC)[5], combined with two gating techniques, partial gating and context-dependent gating (XdG)[6]. We reproduced and tested the performance of the various combinations of the existing methods with different datasets (MNIST, CIFAR[3], and Fashion-MNIST[1]), on different hardware platforms (NVIDIA GTX GPUs and newer version of RTX GPU, and AMD Ryzen CPU). On top of the existing methods, we implemented data augmentation which mixes both MNIST and Fashion-MNIST, as well as new training schemes which let the model review its previously learned dataset to consolidate the models' knowledge of previously learned tasks, which provide further insights on alleviating catastrophic forgetting.

The rest of the paper is organized as follows. From section 2.1 to section 2.3, we briefly introduced four existing methods regrading alleviating catastrophic forgetting and some details of our methods implementations. In section 2.4, experiments using four different processing units were carried out, comparisons and analysis were made based on experimental results. In section 2.5, we tested the existing methods on a new image dataset, Fashion-MNIST, compared the performance with what we obtained with MNIST, and analyzed the possible reasons behind the performance difference. In section 3, we proposed data augmentation methods like datasets mixing, as well as new training schemes including training with review and previous tasks, trying to improve the performance of existing methods with Fashion-MNIST. Results of various experiments regarding above methods were carried out and discussed. In section 4, we drew conclusions based on our results and observations. Lastly, in sections 5, each group member's role in this project was listed.

2 PRIOR WORK

Numerous previous research addressed catastrophic forgetting and presented sound improvements to this phenomenon in several benchmark datasets such as MNIST, CIFAR, ImageNet[7], and etc. In this research, we specifically investigated four combinations of various alleviation techniques. The techniques consist of two synaptic stabilization methods, synaptic intelligence (SI), elastic weight consolidation (EWC) and two gating methods, partial gating, full context-dependent gating (XdG). Briefly, SI works by calculating the correlations between gradient of loss function and parameter updates during training, while EWC focuses on approximating the effects on network output that is caused by changes to parameters. In terms of gating methods, partial gating randomly chooses 50% of hidden units in the network and multiplies them by 0.5 (partial gated), while leaving the other 50% units unchanged. XdG is similar to partial gating, but variant by randomly choosing X% (X = 80 in our testing cases) of hidden units and then multiplying by 0 (fully gated) while leaving the other (1-X)% units unchanged.

We further reproduced the experiments of above existing methods using different hardware setups to investigate the differences between these methods regarding both model accuracies and runtime performance.

2.1 Synaptic intelligence

Before diving into the techniques, it is important to note the essential cause of catastrophic forgetting. When a model trains on task x, it is trying to find the best learning trajectory that maps the inputs to minimal loss, or plainer contexts, minimizing the loss function of task x, L_x. But this operation has no knowledge of any L_y, where $y < x$, thus will inevitably increase the costs of these previous tasks.

Synaptic intelligence (SI) introduced a modified loss function to prevent large changes to important parameters in previous tasks. The loss function of a new training task was modified using the correlations between gradient of loss functions and parameter updates during prior trainings. In other words, each model parameter has been assigned with an associated weight based on how much influence it would have on the previous task's loss function. Model parameters with large impacts will be restricted to change when training with new tasks.

While this technique may not be the most innovative of the time, as we will see later that elastic weight consolidation (EWC) also utilized similar approach, synaptic intelligence used online resources to calculate the entire learning trajectory of each task, which could further improve the modified loss function.

2.2 Elastic weight consolidation

Elastic weight consolidation (EWC) was built on an old concept that for a machine learning model to achieve a specific performance of a given task, the required model parameters are not unique but fall into a specific set [8]. Therefore, the main concept of EWC is taking into account of the relationships between a set of parameters and the previous loss functions. Unlike synaptic intelligence which assigns weights to each individual model parameter, EWC adds an additional quadratic loss function regarding the whole set of model parameters between the old and new training tasks, and pulls the current parameters as closed to its original values as possible, while changing to fit the new tasks. The amounts of the pulls are proportional to the weights' importance in previous trainings.

Instead of calculating the entire learning trajectory like SI does, EWC relies more on the Fisher information metric at the final parameter values to make modification to loss functions, which can only be calculated after the end of each task training. But again, both techniques temper network parameters by the posterior distribution of final parameters in previous tasks.

2.3 Gating

While the previous two techniques enforce constraints on parameter changes and loss functions, the gating methods make modification on network structures. The main idea was inspired by the brain, which can activate highly non-overlapping sets of dendritic branches when switching between tasks. This technique constrains the power of some neurons in the network to prevent the model from altering the whole parameters set. In this research, we focused on partial gating, and context-dependent gating (XdG).

Before each task, partial gating randomly chooses 50% of hidden neurons and multiplies them by 0.5. The identity of the gated neurons will remain fixed during training and testing for a specific task. As for context-dependent gating, it randomly chooses X% (X = 80 in our experiment) hidden neurons and multiplies them by 0, which fully gates the selected neurons.

Because SI and EWC do not modify the neurons, it is therefore possible to combine the gating methods with these synaptic stabilization methods, which became the methods investigated in [6].

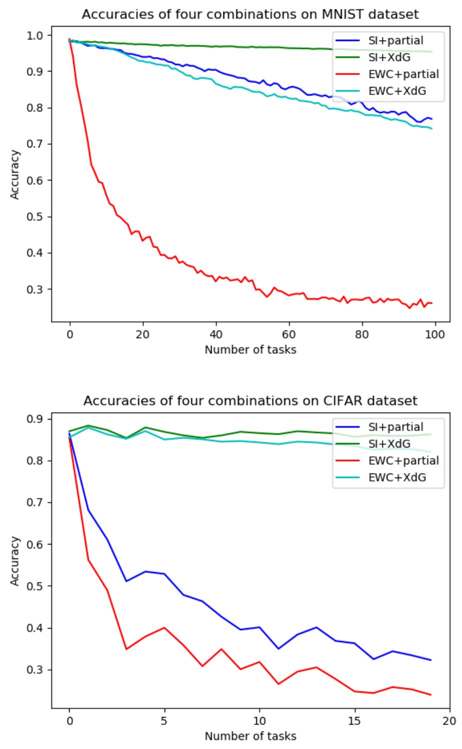

Figure 1: Training accuracies from all methods with MNIST and CIFAR dataset

In the next section, we performed the experiments using several different processing units to observe numerous performance measures.

2.4 Training with different hardware settings

Four different models consist of combinations of the methods introduced in Section 2.1, 2.2 and 2.3 are tested. More specifically, the combinations of methods are SI + Partial gating; SI + XdG; EWC + Partial gating; and EWC + XdG. We referred to implementations published by the original authors of these methods, and tested the performance regarding both machine learning (accuracy) and system (running time and hardware utilizations) concerns. The methods were ran on the TensorFlow framework, with various hardware setups, include AMD Ryzen Threadripper 2950X (CPU), NVIDIA GeForce RTX 2080 (GPU), NVIDIA GeForce GTX 970m (GPU), and NVIDIA GeForce GTX 1050 (GPU). Specifications of these processing units are listed in Table 1.

In terms of the machine learning side, as shown in Figure 1, similar accuracies over 100 (MNIST) and 20 (CIFAR) tasks as reported in the paper had been achieved. It is obvious that SI+XdG has a clear advantage in overcoming catastrophic forgetting, at least with MNIST and CIFAR dataset. This serves as a starting point for testing on other datasets. As SI-XdG is the better-performed combination, new dataset, that shall be seen soon in section 2.5, will be tested with focus mainly on SI-XdG.

On the system side, with the processing units specified in Table 1, we recorded the running times and hardware utilizations of the proposed methods during the training periods with both MNIST (100 tasks) and CIFAR (20 and 50 tasks) datasets. The running times of these models on each hardware specification are shown in Table 2 and 3. Notice that while running on CIFAR dataset, an additional training which is consist of convolutional layers is required, whose training time is shown in Table 6, and not included in Table 3. During our testing with CPU, considering the extremely bad performance when training convolutional layers on a CPU, we used GPU to do all the convolutional training and then let CPU do the rest of the training job. Hardware utilizations during the training periods are shown in Table 4 and 5, which correspond to MNIST and CIFAR dataset respectively.

As for the XdG experiments, notice that we chose 80% of hidden neurons and put them into inactivity. We implemented this by simply multiply them by 0. Thus, the same amount of multiplication operations would be carried out during training compared to no-gating scenario. As a result, we can expect the same runtime performance of XdG compared to that of partial gating. Experimental results as shown in Table 2 and 3 comply with such expectation.

On the one hand, as we expected, the performance of a CPU is much worse than that of a GPU. We observed that during the training process, the AMD 2950X CPU maintained a 70% utilization for all 16 cores. However, it still performs around 30 times slower than RTX 2080. On the other hand, the higher the compute capability of a GPU, the faster it takes to complete the training tasks, which can be easily explained that RTX 2080 has significantly more CUDA cores, higher clock speed, and higher memory bandwidth compared to GTX 970m and GTX 1050 according to Table 1.

The team also observed that RTX 2080 maintained a lower GPU utilization than that of GTX 970m and GTX 1050 during training. We believe that the GPU utilization under deep learning jobs heavily depends on the size and layout of neural networks, training batch sizes, and preprocessing. As for the MNIST case, it requires only a relatively small and straightforward neural network structure to reach an ideal prediction accuracy. As for the CIFAR case, we simply designed four convolutional layers with filters up to only 64. Apparently, RTX 2080's 2944 CUDA cores cannot be fully utilized under such computations.

Surprisingly, we found that GTX 1050 performs slightly better than GTX 970m considering that GTX 1050 has around 50% fewer CUDA cores. We can loosely conclude that clock speed plays an important role when it comes to CUDA core performance. We also tested this by overclocking RTX 2080 to an appropriate and stable state, and as a result we gained a speed boost of around 5% to 10% for each task. Besides, the utilization of GTX 1050 had a major fluctuation varying from 60% to 90% during training. One potential bottleneck for causing this problem is the PCI Express bus usage between the CPU and the GPU, when loading the image data to the GPU in each task.

2.5 Training with Fashion-MNIST

As we observed in Figure 1, SI+XdG barely suffered any catastrophic forgetting effect. It was able to keep the high testing accuracy on the first task, even though the model had been training with permuted-version of the initial task data for a large number of sequential tasks. This ability persisted during the whole 100-task period with MNIST dataset and 20-task period with CIFAR dataset. However, considering that MNIST dataset, which only includes simple digits data, and CIFAR dataset, where permutations of an unique object is still quite unique, we believe that the current testing datasets have their own limitations. More specifically, for MNIST, we argue that although some permutations of a digit number may imply another, for example 6 is a permuted version of 9, considering that it is not a common case, it is not fully convincing that SI+XdG would not fail on other datasets at all. Therefore, we hoped to test the method on a new dataset, which is structurally similar to MNIST, but provides slightly harder images to be recognized. With this intention, Fashion-MNIST became our best choice. It sticks to the 10 classes 70000 gray-scale images in the size of 28x28 as in the original MNIST, but poses a more challenging classification task than the simple MNIST digits data.

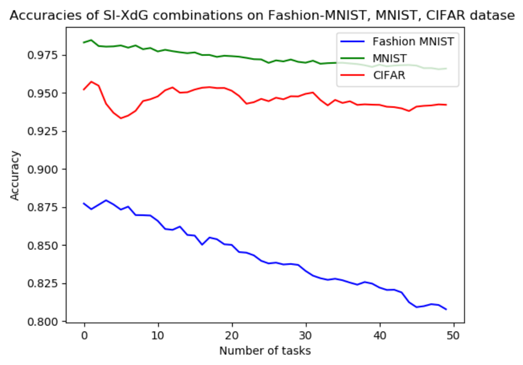

Figure 2: Training accuracies of SI+XdG on Fashion-MNIST

Fashion-MNIST was loaded and preprocessed in the same way as what had been done with MNIST. Since they are structurally the same, we kept the same parameters involved in both SI and XdG algorithm. Figure 2 depicts the accuracy of the first task in Fashion-MNIST, after sequentially training the same model with permutations of the initial data for 50 tasks. It was plotted along with MNIST and CIFAR results for comparison purposes. Notice that we have set the number of training tasks from 100 to 50 for MNIST and 20 to 50 for CIFAR.

Having a closer look at the results, we can observe that, SI+XdG could no longer keep the model away from being influenced by catastrophic forgetting. Even though the curve goes down linearly with a steady rate, instead of an inverse exponential decay as what EWC+Partial gating performed in Figure 1, which means that SI+XdG still played an important role during the whole training period, it is indisputable that SI+XdG alone is no longer sufficient for completely and successfully preventing the model from catastrophic forgetting.

Considering that different methods might have their best performance in different datasets, it is wise not to completely neglect all the other three methods from being tested with Fashion-MNIST. Among them, we picked EWC+XdG, because it is the second best

Hardware	AMD 2950X	NVIDIA 2080	NVIDIA 970m	NVIDIA 1050
Cores	16-Core, 32-Thread	2944 CUDA Cores	1280 CUDA Cores	768 CUDA Cores
Clock speed	3.50 GHz	1815 MHz	924 MHz	1392 MHz
RAM	32GB DDR4 quad-channel	8GB GDDR6	3GB GDDR5	4GB GDDR5
Mem Bandwidth		448 GB/s	120 GB/s	112 GB/s

Table 1: Hardware specifications

	SI + Partial (s)	**SI + XdG (s)**	**EWC + Partial (s)**	**EWC + XdG (s)**
AMD 2950X	86968.79	83086.15	100617.14	97141.83
NVIDIA 2080	4385.30	4180.11	5040.25	5050.17
NVIDIA 970m	13186.42	12570.47	15804.55	15258.65
NVIDIA 1050	12982.68	12371.44	15894.22	15052.79

Table 2: training times of each model with MNIST dataset on different hardware

	SI + Partial (s)	**SI + XdG (s)**	**EWC + Partial (s)**	**EWC + XdG (s)**
AMD 2950X	9299.58	9262.67	12064.11	12030.45
NVIDIA 2080	339.72	331.86	532.58	531.92
NVIDIA 970m	1048.93	981.48	1801.52	1691.33
NVIDIA 1050	957.33	951.28	1637.98	1555.11

Table 3: training times of each model with CIFAR dataset on different hardware

	SI + Partial (%)	**SI + XdG (%)**	**EWC + Partial (%)**	**EWC + XdG (%)**
AMD 2950X	70	69	69	70
NVIDIA 2080	62	63	60	60
NVIDIA 970m	85	84	82	82
NVIDIA 1050	85	82	82	83

Table 4: CPU/GPU utilizations during each training with MNIST dataset

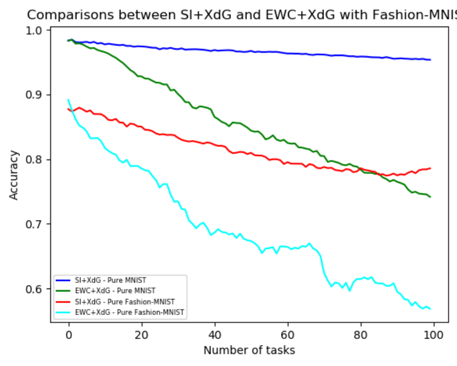

Figure 3: SI+XdG vs EWC+XdG on Fashion-MNIST and MNIST

method based on the results shown in Figure 1. We tested it with Fashion-MNIST and depicted along with MNIST results for comparison, the results are shown in Figure 3.

It is no surprise that, SI+XdG is still the better method than EWC+XdG in terms of alleviating catastrophic forgetting with Fashion-MNIST. And interestedly, we noticed that the resulting patterns that two methods generated on two datasets are very similar. For each method, the difference in terms of accuracy drop between the two datasets are virtually consistent. In other words, the resulting pattern of one dataset can be generated virtually by translating the resulting pattern of the other dataset. Although it does not make sense to translate anything in terms of accuracy, it help illustrate that both methods fail a further same amount of degree when changing training dataset from MNIST to Fashion-MNIST.

After completing the experiments with Fashion-MNIST, we can conclude that, while the current method, SI+XdG, is still better capable of preventing catastrophic forgetting effect, it is not performing well enough for Fashion-MNIST. Therefore, we propose data augmentation method along with new training schemes to first better assess the accuracy decay that happens with Fashion-MNIST, and then improve on top of the current method. These are going to be illustrated with more details in Section 3.

3 NEW TRAINING SCHEMES

From the results shown in Figure 1, it is obvious that, among the four combinations of methods, SI+XdG produced the best results

	SI + Partial (%)	SI + XdG (%)	EWC + Partial (%)	EWC + XdG (%)
AMD 2950X	69	70	70	71
NVIDIA 2080	56	56	55	54
NVIDIA 970m	79	77	76	75
NVIDIA 1050	86	85	85	86

Table 5: CPU/GPU utilizations during each training with CIFAR dataset

	NVIDIA 2080	NVIDIA 970m	NVIDIA 1050
Conv training time (s)	1297.44	4793.38	4413.14

Table 6: convolutional layers training times with CIFAR dataset on different hardwares

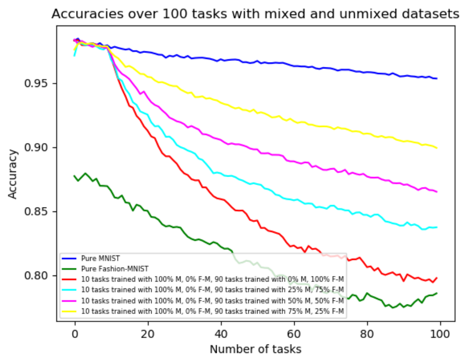

Figure 4: Training accuracies of SI+XdG on all datasets

for all tested datasets. However, as shown in Figure 2 and further discussed in Section 2.5, although SI+XdG is better than other existing methods, it still suffered a severe catastrophic forgetting when training with Fashion-MNIST. To better evaluate how SI+XdG performs with MNIST and Fashion-MNIST, we prepared the training data to be a mix from both MNIST and Fashion-MNIST datasets, instead of using permutations from one dataset alone. Experiments regarding this is explained in more details in Section 3.1.

Moreover, we also want to specifically address the issue that SI+XdG method suffers from Fashion-MNIST dataset, and further improve the results. Inspired by the learning scheme that human beings possess, where frequent reviewing helps keep a longer memory after his/her first exposure to the new knowledge, two new training schemes, which involves different types of reviews, have been proposed and tested. Details of scheme implementations and testing outcomes are presented in section 3.2 and 3.3.

3.1 Testing SI+XdG with mixed data

There was no doubt that training with Fashion-MNIST would affect SI+XdG's performance in terms of overcoming catastrophic forgetting. However, would this influence appear when only a specific percentage of Fashion-MNIST exists in the whole dataset? What is the relationship between this percentage and the performance decay; is it proportional? inverse-exponential or no clear mathematical correlations at all? These questions remained unanswered. On the other hand, there is a real-world-related reason of why mixing the dataset as disturbing tasks is important when evaluating algorithm's ability to alleviate forgetting; in most cases, catastrophic forgetting happens when a model is being trained with data that is independent and different to the initial training data, or a mix of multiple datasets which may or may not include the initial training data. Therefore, testing an algorithm with disturbs consist of only permutations of itself is clearly insufficient.

As it has been observed that there exists a big difference in terms of accuracies when training with MNIST and Fashion-MNIST datasets, the experiments were designed to first generate mixed datasets with MNIST and different percentages of Fashion-MNIST. We hoped to find out the trends of how accuracy dropped with respect to the change of percentage.

As shown in Figure 4, during the first 10 tasks, the model was trained with 100% MNIST dataset and its permutations, which explains why the accuracies all collapse during the first 10 tasks. After that, training data of each task has been changed to a certain percentage of mixture between both MNIST and Fashion-MNIST. We observe that, as expected, as the percentage of Fashion-MNIST data gets larger, catastrophic forgetting becomes more severe, and converges to the pure Fashion-MNIST case more. More specifically, the accuracy decay is proportional to the change of percentage; there was a 8-10% drop in accuracy when 25% more Fashion-MNIST data was added to the mixed training data.

3.2 Training with review

We found out, from section 3.1, that if we treated MNIST as initial-tasks data, Fashion-MNIST could be a great fit as new-task dataset which disturbed the model's ability to perform well on MNIST after sequentially training with lots of Fashion-MNSIT tasks. Fashion-MNIST provided a clear and distinguishable decay in terms of accuracy even at a low percentage of mix (25%). Moreover, its percentage increase is nearly linearly proportional to the accuracy decay, which provides easier benchmarks when comparing training scheme outcomes between different mixes.

To make the SI+XdG better overcome catastrophic forgetting when disturbed by Fashion-MNIST datasets, we proposed two new training schemes, which are explained in section 3.2.1 and 3.2.2.

3.2.1 Different review frequencies.

Benchmark experiments include training with 100% MNIST for the entire 100 tasks, and training with 100% Fashion-MNIST for the entire 100 tasks (top and bottom line in Figure 5). All other

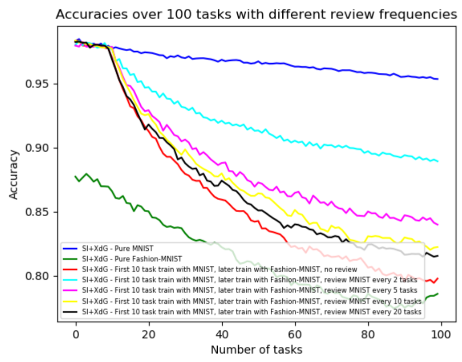

Figure 5: Training accuracies of SI+XdG on Fashion-MNIST with different review frequencies

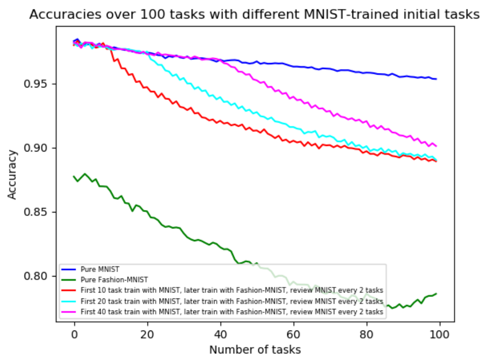

Figure 6: Training accuracies of SI+XdG on Fashion-MNIST with different initial task lengths

experiments had the same basic training scheme; the model was trained with 100% MNIST data for the initial 10 tasks and later trained with 100% Fashion-MNIST dataset for the rest 90 tasks. The difference is the review frequencies, which reviews permuted MNIST data with frequencies ranging from every 2 tasks to every 20 tasks.

As shown in Figure 5, in general the results align with the intuitive guess that less suffering in accuracy happens when reviewing more frequently. However, it is surprising that, the largest accuracy drop happens when going from pure-MNIST benchmark to reviewing every 2 tasks, and then from every 2 tasks to every 5 tasks. It is unexpected because, obviously, reviewing every 2 or 5 tasks are both very high frequencies, which intuitively "should not" cause a large accuracy loss. However, recalling that what both SI and XdG are trying to achieve is to prevent the model parameters being changed by a single task. Therefore, it in fact makes sense that a review task would not help the model regain what it had lost during the training with Fashion-MNIST. Further, this suggests that training with reviews could be added on top of SI and XdG to better overcome catastrophic forgetting, but is not quite efficient since it requires a very high review frequency yet still would suffer from severe accuracy loss.

3.2.2 *Different initial task lengths.*

Another intuition we have about human memory is that the materials which have been studied longer will also be remembered longer and harder to forget. Inspired by this, we designed a set of experiments to find out if this is also true regarding model training and catastrophic forgetting. This is in fact meaningful in overcoming catastrophic forgetting. Because if we found out the relationship between the initial training length and the later forgetting rate, we can help overcome catastrophic forgetting by applying a calculated longer training period to a specific task so that we can estimate its forgetting rate to fall into our acceptable range, while using the same techniques we have currently, like SI+XdG.

The experiments were set up similarly to those in section 3.2.1. The benchmark experiment are still training with 100% MNIST for the entire 100 tasks and training with 100% Fashion-MNIST for the entire 100 tasks. All others were trained with 100% MNIST data for the initial X tasks, and later trained with 100% Fashion-MNIST dataset for the rest $(100 - X)$ tasks, where X represents the number of initial training length. Note that no review happens in any of these experiments. We ran the experiments with $X = 10, 20$ and 20.

The results are quite unexpected, as shown in Figure 6, even if the three experiments had different starting times with Fashion-MNIST tasks, they seemed to have the same level of knowledge and performance about the initial task after 100 total number of tasks. In other words, the experiment which had been trained with only 60 Fashion-MNIST tasks (pink line in Figure 6) suffered a similar amount of information loss regarding initial task as the model which had been trained with Fashion-MNIST for 80 (cyan line in Figure 6) or even 90 (red line in Figure 6) tasks. This is totally on the opposite of our intuition, where training initial task longer would help overcome forgetting. In fact, this suggests that there is no need to train a model with dataset that has been trained before for excessive time in the hope of making the model more stable to disturbs.

3.3 Training with previous tasks

While the previous two subsections investigated different scenarios of mixing MNIST with Fashion-MNIST, in this section we hoped to test out the self-review of Fashion-MNIST.

The team tried to improve the state of art performance on Fashion-MNIST by mixing a portion of data from previous tasks with each current task, so that for each task t, it will contain X% of current data and (1-X)% of either data from task $t - 1$ or a random task $< t$.

We expected a clear boost to accuracy because reviewing old tasks seemed like an intuitively correct approach for potential improvements. The green line in Figure 7 represents the accuracy of training each task N with 80% of current data and 20% of data from task N - 1. Even though the difference in accuracy between training with and without self-review is less than 1%, the self-review version started to gain a small margin after 50 tasks. The team neglected the last few fluctuations of training without self-review because the pattern was not reproducible with other dataset and they do not correspond with our other observations.

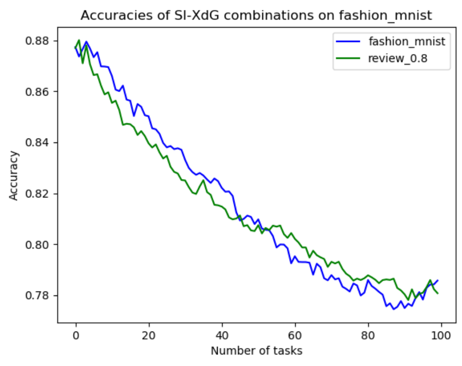

Figure 7: Training accuracies of SI+XdG on Fashion Mnist with 20% Review

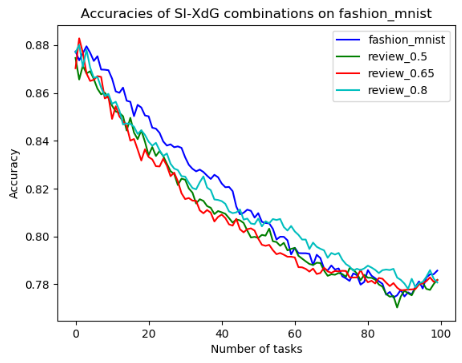

Figure 8: Training accuracies of SI+XdG on Fashion Mnist with different review portion

After the team found that self review could alleviate catastrophic forgetting even more, we tested out different proportion of reviewing. In Figure 8, we compared training without self review to training with 50%, 35%, and 20% data from task N - 1 for each task N. Reviewing with 20% old data still yield the best results from our experiments. We suspect the main reason is that the network still require enough coherence within tasks to perform properly, while a small portion of review indeed prevents the network from forgetting slightly.

Last, the team implemented a version of self review in which for each task N, it will review a random previous task instead of always task N - 1; the intuition was to make the self review more generic. In Figure 9, we compared training with 50% data from task N - 1 and training with 50% data from a random task. We can observe that the random self review works slightly better than non-random self review, while we also neglected the unstable pattern.

From the experiments we performed in this subsection, it is clear that randomly self reviewing a small portion of previous data in each task could further increase accuracy in sequentially learning tasks.

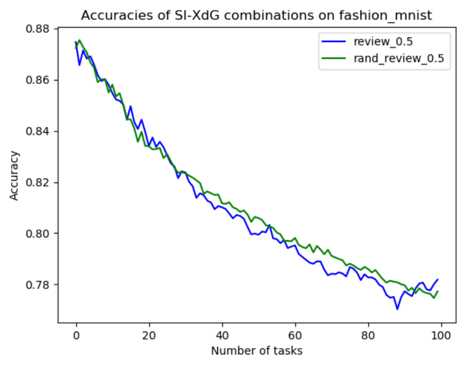

Figure 9: Training accuracies of SI+XdG on Fashion Mnist with random task review

4 CONCLUSIONS

In the first half of the research, the team performed deep investigation on four existing methods that could alleviate catastrophic forgetting, namely Synaptic Intelligence, Elastic Weights Consolidation, Partial Gating, and Context Dependent Gating. We tested out their performance using four processing units, AMD 2950X, NVIDIA 2080, NVIDIA 970m, and NVIDIA 1050, to gain further insights on the methods and the roles of different performance specifications such as GPU utility and memory usage.

Both previous work[6] and our experiments showed that Synaptic Intelligence combined with Context-Dependent Gating yield the best result on overcoming catastrophic forgetting tasks with MNIST and CIFAR. We later found the combination also worked the best with Fashion-MNIST, though with significantly lower accuracy than with MNIST, which had exactly the same data shape.

This is the main reason why we proceeded with three data mixing experiments to further study their behaviors. By mixing MNIST with different percentages of Fashion-MNIST, we not only found the relationship between such percentage and the performance decay of existing methods regarding alleviating catastrophic forgetting, but also provided a better estimate of how the existing methods would perform in a real-world scenario, which was lack in their original experiments. We later proceeded training with two review schemes, which led to the conclusion that neither reviewing initial-task data or excessive training with initial task would help much regarding overcoming catastrophic forgetting. Last, we found that by mixing a portion of previously trained Fashion-MNITS into every training task, the accuracy gained a small margin over the case using pure Fashion-MNITS without reviewing. Randomly choosing one of the previous tasks to review each time also performs slightly better than always reviewing the previous one task.

These phenomenons lead to our suggestions regarding future work. Relatively small amount of data shuffling and augmentation test cases have been tried in previously research; our study shows that there are merits in further processing the input data to allow data reviewing, though more rigorous work will be required to find the most reasonable and effective data shuffling techniques for this problem. In addition to data augmentation, while the current gating methods show decent results on alleviating catastrophic forgetting, the way they choose gated neurons are pure random, and we believe there should be non-random operations that could further boost the effectiveness of the gating methods.

5 EACH PERSON'S ROLE

(1) Zhuokai Zhao – Investigated existing methods and the associated Github repositories, reproduced results from existing methods, and implemented new training schemes introduced in Section 3.1 and 3.2.
(2) Ping-Jung Liu – Investigated existing methods and the associated Github repositories, reproduced results from existing methods, and implemented new training scheme introduced in Section 3.3.
(3) Nanqinqin Li – Reproduced results from past research, analyzed hardware performance between various processing units.

ACKNOWLEDGMENTS

We sincerely appreciate Professor Stevens, Professor Foster, and Sam Nickolay's teaching and guidance throughout the course.

REFERENCES

[1] Han Xiao, Kashif Rasul, Roland Vollgraf, Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms, arXiv, 2017
[2] Yann LeCun, Corinna Cortes Mnist handwritten digit database, 1998
[3] Alex Krizhevsky, Vinod Nair and Geoffrey Hinton, CIFAR-10 (Canadian Institute for Advanced Research), http://www.cs.toronto.edu/ kriz/cifar.html
[4] Zenke F, Poole B, Ganguli S, Continual Learning Through Synaptic Intelligence, International Conference on Machine Learning, 2017
[5] Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A et al, Overcoming catastrophic forgetting in neural networks, Proceedings of the National Academy of Sciences, 2017
[6] Nicolas Y. Masse, David J. Freedman, Gregory D Grant, Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization, 2018
[7] Deng, J. and Dong, W. and Socher, R. and Li, L.-J. and Li, K. and Fei-Fei, L., ImageNet: A Large-Scale Hierarchical Image Database, 2009, http://www.image-net.org/papers/imagenet_cvpr09.bib
[8] Robert H. Nielsen, Theory of the backpropagation neural network, Proceedings of the International Joint Conference on Neural Networks, volume I, pages 593âĂŞ605, Piscataway, NJ: IEEE, 1989